Project Brief: User Interface for Multi-Source Document Integration with AI

Project Overview

This project involves the creation of a user interface (UI) that allows users to input multiple sources of documents as data sources. The interface will be connected to OpenAI and Claude AI via APIs, enabling users to interact with the system through prompts. The UI will then compile information from all the connected data sources and provide a coherent knowledge output.

Objectives

1. Develop a User-Friendly Interface: Create an intuitive and user-friendly interface that can easily accept and manage multiple document sources.
2. Integrate AI APIs: Connect the interface with OpenAI and Claude AI via APIs to facilitate AI-driven interactions.
3. Document Handling: Enable the interface to accept and process documents in various formats (e.g., PDF, Word, Excel, etc.).
4. Prompt Interaction: Allow users to interact with the AI models through customizable prompts.
5. Knowledge Compilation: Design a system that compiles information from all the data sources and presents a cohesive output.
6. Scalability and Flexibility: Ensure the interface is scalable and flexible to accommodate future expansions, such as additional AI models or data sources.

Key Features

- Multi-Source Document Input: Users can upload and manage documents from various sources within the interface.
- AI Interaction: The UI will enable users to interact with OpenAI and Claude AI through prompts, receiving responses based on the compiled information.
- Data Compilation: The system will gather and compile data from all sources, providing a summarized or detailed knowledge output depending on user preferences.
- Customizable Prompts: Users can customize prompts to tailor the AI responses to their specific needs.
- User Access and Permissions: Implement user access controls to manage who can upload, modify, or interact with the data sources.
- Output Formats: Provide multiple formats for the knowledge output, such as text summaries, reports, or exportable files.

Technical Requirements

- Frontend Development: Develop the UI using modern web technologies (e.g., React.js, Angular, or Vue.js).
- Backend Integration: Create a backend system to manage document storage, API connections, and data processing (e.g., using Node.js, Python, or Java).
- API Integration: Integrate with OpenAI and Claude AI using their respective APIs, ensuring secure and efficient communication.
- Data Storage: Implement a robust data storage solution (e.g., a cloud-based database) to handle the documents and generated knowledge outputs.
- Security Measures: Ensure the system is secure, with encryption for data in transit and at rest, and implement authentication/authorization protocols.

Success Criteria

- User Satisfaction: Positive feedback from users regarding the ease of use and effectiveness of the interface.
- System Performance: The interface should perform efficiently without lag, even with multiple document sources.
- Accurate Knowledge Outputs: The compiled knowledge outputs should be accurate and relevant, meeting user expectations.
- Security Compliance: The system must adhere to all relevant security standards, ensuring the protection of sensitive data.